Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(4)

AD-A214 680

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER HIG Contribution 1852	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Rapid generation of synthetic seismograms in layered media by vectorization of the algorithm		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Robert A. Phinney, Robert I. Odom, and Gerard J. Fryer		8. CONTRACT OR GRANT NUMBER(s) N00014-80-0924
9. PERFORMING ORGANIZATION NAME AND ADDRESS Hawaii Institute of Geophysics 2525 Correa Road Honolulu, Hawaii 96822		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Ocean Sciences & Technology Division Bay St. Louis, MS 39520		12. REPORT DATE December 1987 13. NUMBER OF PAGES 9 pages
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research, Branch Office 1030 East Green Street Pasadena, CA 91106		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release: Distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Published in Bull.Seismol. Soc. Amer., v. 77, no. 6, p.2218-2226, 1987

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Synthetic seismograms
layered media
modeling
sedimentary rock
vectorization

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The theory and practice of computing synthetic seismograms in layered media are now quite well understood, and are nicely set forth in the monograph by Kennett (1983) and the textbook by Aki and Richards (1980). In applications, these methods can still be unsatisfactorily slow.

It may be necessary to work with models having a large number of microlayers. Modeling of reflection seismology data in typical sedimentary rocks may commonly require more than 100 layers. Many situations having a smooth velocity gradient and need many microlayers to model the gradient, when, for one reason or another asymptotic methods are not suitable. For example, if we are studying wide-angle or long-range propagation in the top 600 m of a sedimentary section, we need to model the pronounced near-subbottom material gradients due to compaction as well as particular layers which are high or low velocity, density, or Q anomalies with respect

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) to the reference gradient.

In seeking to determine the variation of material properties in a sedimentary pile, we may start with some generalized reflection data set, containing reflected and refracted signals produced by an impulsive point source and recorded at a number of sensors in the water layer at various offsets from the source. Any true inversion procedure, regardless of details, must compare the observed data with synthetic data *for a sequence of models in order to obtain and characterize models which fit* the data within some prescribed limits. We thus seek a forward modeling procedure for arbitrary layered models which is fast enough that many repetitions are possible in a reasonable time.

X

A-1 20

Bulletin of the Seismological Society of America, Vol. 77, No. 6, pp. 2218–2226, December 1987

RAPID GENERATION OF SYNTHETIC SEISMOGRAMS IN LAYERED MEDIA BY VECTORIZATION OF THE ALGORITHM

BY ROBERT A. PHINNEY, ROBERT I. ODOM*, AND GERARD J. FRYER

The theory and practice of computing synthetic seismograms in layered media are now quite well understood, and are nicely set forth in the monograph by Kennett (1983) and the textbook by Aki and Richards (1980). In applications, these methods can still be unsatisfactorily slow.

It may be necessary to work with models having a large number of microlayers. Modeling of reflection seismology data in typical sedimentary rocks may commonly require more than 100 layers. Many situations having a smooth velocity gradient and need many microlayers to model the gradient, when, for one reason or another asymptotic methods are not suitable. For example, if we are studying wide-angle or long-range propagation in the top 500 m of a sedimentary section, we need to model the pronounced near-subbottom material gradients due to compaction as well as particular layers which are high or low velocity, density, or Q anomalies with respect to the reference gradient.

In seeking to determine the variation of material properties in a sedimentary pile, we may start with some generalized reflection data set, containing reflected and refracted signals produced by an impulsive point source and recorded at a number of sensors in the water layer at various offsets from the source. Any true inversion procedure, regardless of details, must compare the observed data with synthetic data for a sequence of models in order to obtain and characterize models which fit the data within some prescribed limits. We thus seek a forward modeling procedure for arbitrary layered models which is fast enough that many repetitions are possible in a reasonable time.

The procedure that we describe in this paper is based on a reorganization of the inside loops of a conventional reflectivity algorithm to permit vectorization. With this optimization, it is unncessary to be concerned in detail with optimization of the code that generates individual matrix elements from their algebraic expressions, a technique adopted by Kind (1976). The vectorized procedure is by itself faster than previously described procedures, simply because practically all redundancy has been eliminated. When implemented on an array processor, however, it achieves additional speed enhancement of about 20, by making effective use of the vector hardware. We have implemented this procedure on a Data General Nova 3, and on a Perkin-Elmer 3230, with a CSPI MAP 300 array processor, in both *Fortran* and *C*. In particular, we have used a *slowness* method, with the Haskell-Dunkin 6×6 minor matrices for construction of the reflectivity (Haskell, 1953; Dunkin, 1965). An optimal code in terms of stability and computational overhead would probably use the Kennett 2×2 reflection-transmission matrices. We will take as a basis for most of this discussion the monograph by Kennett (1983) and not repeat much standard material here.

A vectorized version of the WKBJ method has been developed and tested by Chapman and Orcutt (1985), and applied to the inversion of marine refraction data by Shaw and Orcutt (1985). For certain problems requiring summation of many rays to synthesize the signal, or when investigating phases omitted by the WKBJ

* Present address: Geophysics Program AK-50, University of Washington, Seattle, Washington 98195.

e. 1852

theory, the more accurate, but also the more time-consuming reflectivity method would be required. This has been a motivation for developing our algorithm.

The Procedure

Consider an impulsive point pressure source in a layered stack of sediments, with a surface water layer, and the observed acoustic pressure field near the surface. In the slowness/reflectivity method, the acoustic pressure field $P(r, t)$ may be represented by the double integral

$$P(r, t) = \int_{-\infty}^{\infty} d\omega e^{i\omega t} S(\omega) \int_0^{\infty} dp\, p J_0(\omega p r) \Phi(\omega p, p) \tag{1}$$

where $p(= k/\omega)$ is slowness, Φ is the plane wave response function of the layered medium, and $S(\omega)$ is the Fourier transform of the source-time function $S(t)$. Carrying out the integration over ω in (1) gives the exact relation for the solution as

$$P(r, t) = \frac{1}{r} S(t) * D^2 \int_0^{\infty} \hat{\phi}(p, \tau) * \hat{J}_0\left(\frac{\tau}{p} r\right) dp, \tag{2}$$

where $\hat{J}_0(t)$ is the inverse Fourier transform of $J_0(\omega)$, and $\hat{\phi}$ is the inverse Fourier transform

$$\hat{\phi}(p, \tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-i\omega t} S(\omega) \zeta(p, \omega) R_{PP}(\omega p, \omega) \sigma(p, \omega)\, d\omega. \tag{3}$$

Although we consider cylindrical waves, $\hat{\phi}$ may be regarded as the plane wave transient response of the medium with the given source and receiver functions. To generate the plane wave transient response, we need the following components from (3)

$S(\omega)$ = Fourier transform of the source time wavelet
$\zeta(p, \omega)$ = source directivity function ($= 1/i\omega\eta_0$, for a point pressure source)
$R_{PP}(\omega p, \omega)$ = reflectivity or total medium plane wave response function
$\sigma(p, \omega)$ = receiver directivity function ($= 1$, for a pressure receiver).

In this paper, we present a scheme by which the computation of the plane wave transient response $\phi(p, \tau)$ may be optimized for use with an array processor. The evaluation of (2) is a separate question, for which no particularly rapid algorithm has been proposed, in the general case. In the very common instance, however, where the source receiver offset is not close to zero, (2) can be rapidly evaluated by the slant stack

$$P(r, t) = \frac{1}{2\pi} \frac{d}{dt} H^{+} \int_0^{\infty} \hat{\phi}(p, t + px)\, dp \tag{4}$$

(Chapman, 1978; Phinney *et al.*, 1981).

In order to clarify our new algorithm, we summarize the familiar *spectral* and *slowness* methods ($\langle A \rangle$ and $\langle B \rangle$).

$\langle A \rangle$: for ($\omega = 0$ *to* $\omega = \omega_{\max}$)

```
initialize
for ($k = 0$ to $k = k_{\max}$)
  ($\omega$ fixed)
  for (layer = 0 to layer = $N$)
      ($k, \omega$ fixed)
    evaluate layer and interface matrices
    matrix multiplies
  integrate over $k$  (Bessel function evaluation)
integrate over $\omega$.
```

Similarly, for the *slowness* method we have

$\langle B \rangle$: for ($p = 0$ *to* $p = p_{\max}$)

```
initialize
for ($\omega = 0$ to $\omega = \omega_{\max}$)
  ($p$ fixed)
  for (layer = 0 to layer = $N$)
      ($\omega, p$ fixed)
    evaluate layer and interface matrices
    matrix multiplies
  integrate over $\omega$  (Fast Fourier Transform $\rightarrow \hat{\phi}(p, \tau)$)
integrate over $p$  (slant stack)
```

$\langle B \rangle$ is more efficient than $\langle A \rangle$ as it avoids the Bessel function evaluations and the oscillatory integral over k. Both algorithms, however, have essentially the same structure for the innermost loops. Starting from the slowness algorithm $\langle B \rangle$, we recognize the inner loops to achieve a vectorized algorithm, which is an optimum way of generating the plane wave transient response $\hat{\phi}(p, \tau)$.

The principle computational burden in evaluating $\hat{\phi}(p, \tau)$ lies in the evaluation of the response of the medium to plane waves paramaterized by $k = \omega p$ and ω. We are required to evaluate the reflectivity of the stack of layers, as seen at a reference level in layer 0. In the Haskell-Dunkin minor matrix method, which we have implemented, and which we discuss for concreteness, a six-element vector

$$v_N = \mathrm{col}[1, 0, 0, 0, 0, 0]$$

is established in the lower half-space, and analytically continued upward to layer 0 by a sequence of matrix multiplications

$$v_0 = Q_0 Q_1 \cdots Q_{N-1} v_N. \tag{5}$$

(The specific form taken by these matrices depends on the exact starting definitions used in setting up the problem. See the Appendix for the formulation used here.) We then obtain directly the reflection functions by inspection of v_0

$$v_0 = \mathrm{col}[\Delta, -R_{PS}\Delta, -R_{SS}\Delta, R_{PP}\Delta, R_{SP}\Delta, \det R\Delta]. \tag{6}$$

Since the reflection functions arise from v_0 as ratios of elements, the procedure (5) may be modified by multiplying v_n by any complex scaling constant at any stage n, without affecting the result. Each wave propagator matrix Q_n is the product of an interface matrix F_n and a layer crossing matrix E_n

$$Q_n = E_n F_n.$$

We will obtain a fast algorithm by exploiting two properties of these matrices

F_n can be written: $F_n(p)$

E_n can be written (7a)

$$\mathrm{diag}[e^{-\iota\omega d_n(q_n+q_n')}, 1, e^{-\iota\omega d_n(q_n-q_n')}, e^{\iota\omega d_n(q_n-q_n')}, 1, e^{\iota\omega d_n(q_n+q_n')}] \quad (7b)$$

where d_n is layer thickness and q_n, q_n' are P and S vertical slownesses.

We can exploit the properties (7a) and (7b) to convert $\langle \boldsymbol{B} \rangle$ to a vectorizable algorithm, as follows.

1. Loops are nested (outer to inner): loop in p; loop on layer $[n]$; loop on frequency $[m]$.
2. The innermost (frequency) loop is redefined to consist of vector processes, in which the dimension of the relevant complex vectors is the number of frequencies, M, being evaluated. On an array processor, this loop can be implemented as true vector operations, while on an ordinary processor, the vectorized code provides substantial computing advantages over the conventional procedure.
3. Suppose that the code is to be run for M equally spaced frequencies. The six-element complex state vector v is assigned storage for a $6 \times M$ complex vector $\hat{v}$

$$\hat{v}^{(T)} = \begin{bmatrix} v_1^{(1)}, v_2^{(1)}, \cdots, v_6^{(1)}, \\ v_1^{(2)}, v_2^{(2)}, \cdots, v_6^{(2)}, \\ \vdots \\ v_1^{(M)}, v_2^{(M)}, \cdots, v_6^{(M)} \end{bmatrix} \begin{array}{l} \textit{1st frequency} \\ \textit{2nd frequency} \\ \vdots \\ \textit{Mth frequency} \end{array} \quad (8)$$

   This vector is, for each new value of p, initialized with $v_1^{(m)} = 1.0$, for $m = 1, 2, \cdots, M$, and all other elements set equal to zero. The calculations then proceed by carrying out each successive matrix multiplication in (5) for all frequencies by operation on the extended $6 \times M$ state vector $\hat{v}$.
4. Consider the matrix multiplication

$$v_n = E_n F_n v_{n+1} \quad (9)$$

   in which the state vector is analytically continued from the top of the $(n + 1)$ layer to the top of the (n) layer. When we extend v_n to $\hat{v}_n$, by including all frequencies, we need suitable extensions of F_n and E_n. Now, the complex 6×6 matrix F is (7a) independent of frequency. We may thus provide storage for the values of F for the N interfaces, which may be precomputed once for each value of p (the outer loop). The matrix multiplication

$$u = F_n \hat{v}_{n+1}$$

uses the same 36 values of F_n in operating on $\hat{v}_n$ for all frequencies. In an array processor environment, this operation can be set up as a complex convolution of the rows of F_n with $\hat{v}_n$, with a shift of 6.

5. Matrix multiplication by E_n requires that E_n be given extended storage for M frequencies. We treat E_n as a complex six vector, since it is diagonal, and obtain a $6 \times M$ representation similar to (8), we we call $\hat{E}_n$. The matrix multiplication

$$\hat{v}_n = \hat{E}_n u \tag{10}$$

becomes a dot product of two complex $6 \times M$ dimensional vectors.

6. Computation of the values of $\hat{E}_n$ requires some optimization, for the algorithm would seem to demand $6 \times M \times N$ complex values to be computed and stored. In particular, we seek to reduce what appears to be $2 \times M \times N$ complex exponential evaluations. We do this by generating the values for $\hat{E}_n$ recursively, using the first six elements (zero frequency) as starting values. Assign storage for N complex "incrementor" vectors G_n. If

$$s_n^+ = e^{i\Delta\omega d_n(q_n + q_n')} \tag{11}$$

and similarly for s_n^-, then we define

$$G_n = [1/s_n^+, 1, 1/s_n^-, s_n^-, 1, s_n^+]. \tag{12}$$

The collection of G_n depends on p, but not on ω, and may be precomputed once for each value of p, in the outer loop. Consequently, $\hat{E}_n$ is generated by assigning starting values (for zero frequency): $E_{1,2,\cdots,6} = 1.0$, and by using G_n to generate the higher frequency values of $\hat{E}_n$ by complex multiplication

$$\begin{bmatrix} E_m \\ E_{m+1} \\ \vdots \\ E_{m+5} \end{bmatrix} = \begin{bmatrix} G_1 \\ G_2 \\ \vdots \\ G_6 \end{bmatrix} \cdot \begin{bmatrix} E_{m-6} \\ E_{m-5} \\ \vdots \\ E_{m-1} \end{bmatrix} \quad \text{for } m = 7, 13, \cdots, 6 \times (M-1) + 1. \tag{13}$$

The basic computational unit thus consists of the following vector process (which we label $\langle P \rangle$) which has the effect of continuing the system vector $\hat{v}$ from the top of layer $n + 1$ to the top of the layer n, for all frequencies

- Complex convolve the rows of F_n with $\hat{v}_{n+1}$, with a skip of $6 \longrightarrow u$.
- Complex dot product of $\hat{E}_n$ with $u \longrightarrow \hat{v}_n$.
- Generate new vector $\hat{E}_{n-1}$ for the next layer.

These may be implemented as array processor calls or, lacking an array processor, as an inner loop over frequency. $\hat{v}_N, \hat{v}_{N-1}, \cdots$, may all occupy in succession the same storage array.

The algorithm is summarized as $\langle C \rangle$

```
⟨C⟩:   for ($p = p_0$ to $p = p_{max}$)
```

```
⟨compute the reflectivity for this $p$, all $\omega$⟩
initialize $\hat{F}$
initialize $G_n$, for all $n$
for ($layer = N - 1$ to $layer = 0$)
   ⟨continue $v$ from one layer to next⟩
   execute vector process ⟨$P$⟩
      ⟨handles all frequencies⟩
Generate plane wave seismogram by Fast Fourier Transform.
```

One pass through the stack of layers produces the complex reflectivity for all frequencies; the plane wave seismogram is then produced by Fourier transformation.

Incorporation of source and receiver directivity and spectrum into the algorithm is easily done, as is a frequency domain operator embodying $H^{+}(d/dt)$ [equation (4)]. The algorithm may be easily redesigned to use one of the more popular matrix schemes, such as the reflection-transmission matrix scheme of Kennett (1981, 1983), which is wholly free of potential overflow or underflow problems. Incorporation of the effects of a free surface, and of source and receiver at different levels, requires the construction of a response matrix from the reflectivity matrix using methods described by Kennett (1983), but lie outside the inner loop and do not appreciably affect the computational load. In the Haskell-Dunkin method, the computation of the elements of $\hat{E}$ can lead to overflow and underflow at higher frequencies, particularly when the compressional wave function in one or more layers is evanescent. To avoid these problems, we normalize $\hat{E}$ by its largest value; a process which does not affect the reflectivities, as we have remarked.

Speed and storage. The performance characteristics of our implementation are summarized in Table 1. Even though the algorithm is based on formation of long

TABLE 1

Summary of the Performance Characteristics of our Implementation of the Reflectivity Algorithm

Speed and Storage for Algorithm (C)	
Processor:	Perkin-Elmer 3230 with 1k cache, 3mb memory, and floating point hardware
Array processor:	CSPI MAP 300
Operating system:	Unix version VII
Language:	C
Test speed:	50,000 indexed floating point multiply-adds per second
Algorithm:	Tested for 256 frequencies, one value of p
	2.2 sec per layer: 3230, with subscripted array references
	0.6 sec per layer: 3230, with pointers for array access
	30 msec per layer: array processor
Storage:	Given N layers, M frequencies, and using $N = 25$, $M = 256$.

Array	Type	Words	Bytes	
$\hat{v}$	Complex	$6 \times M$	$48 \times M$	12,288
$\hat{P}_n$	Complex	$36 \times N$	$144 \times N$	3,600
$\hat{E}$	Complex	$6 \times M$	$48 \times M$	12,288
G_n	Complex	$6 \times N$	$48 \times N$	600
Total	—	—	—	28,776

vectors, from Table 1 we see that the storage actually required is still well within the capability of commonly used computers. Use of pointer access methods, which are available in the *C* language (or in assembler patches under Fortran), provided a speed improvement factor of 3.6. The extra factor of 20 obtained by use of the array processor is clearly a major factor in making this algorithm one that might be used in repetitive situations. While any code may be marginally improved by close attention to eliminating unnecessary operations, such as multiplication by 1 or 0, this procedure appears to be nearly optimum. Function evaluations (exponentials and square roots) have only a trivial effect on the total time, and matrix element evaluations have been removed from the innermost loop, as far as possible. We judge that any further order of magnitude improvement in performance will be through the use of suitable hardware, such as a state-of-the-art (250 mflops) array processor.

Use in an inversion process. We seek particularly high speeds to facilitate the inversion of reflection-refraction data. In an inversion procedure, one is asked to take a single data set and to generate a potentially large number of synthetic data sets for iteration to a solution and for assessment of the class of models which fit the data in some sense. An optimum data set for this purpose would consist of sufficiently many receivers in a radial array, so that spatial aliasing and windowing effects are unimportant. It would then be possible to generate an accurate representation of the plane wave decomposition of the data. The inversion process would then generate synthetic data for a very restricted subset of slownesses, which contains most of the independent information required to fit the data. Suppose, for example, that five plane wave seismograms are being generated per pass at a model, and 2 sec are required per slowness; then 360 models can be evaluated per hour. If it is desired to generate the spatial representation $\phi(r, t)$, then over 200 plane wave seismograms are needed as input into a reasonably accurate slant stack, and only nine models per hour can be generated.

Discussion

Our purpose in needing a fast algorithm is to be able to explore a large variety of models, both globally and incrementally. At this point, we discuss these models in the sense that we are simulating a laboratory experiment and trying to develop a more thorough understanding of the wave propagation phenomena which characterize the sea-bottom sediment column. This understanding is a prerequisite to applying the modeling procedure in an inversion process, which will require judicious choice of procedures for subselecting data and model parameters.

At this point we have no idea whether the speed achieved is fast enough. With the array processor, computation speeds which deliver tens of models per hour are certainly insufficient for even a restricted Monte Carlo procedure; perhaps more orderly inversion algorithms can be established which can make good use of this capability. With modern supercomputers and/or modern superarray processors, however, hundreds of models per hour are now feasible. Whether the application justifies that kind of expense remains to be seen.

Acknowledgments

This research was supported by the Office of Naval Research, under Contract N00014-80-K-0924, and by research funds of the Department of Geological and Geophysical Sciences, Princeton University. Computer facilities for the final preparation of this manuscript were generously provided by Ron Merrill and Bob Crosson of the Geophysics Program at the University of Washington.

References

Aki, K. and P. G. Richard, (1980). *Quantitative Seismology*, W. H. Freeman and Co, San Francisco, California.

Chapman, C. H. (1978). A new method for computing synthetic seismograms, *Geophys. J. R. Astr. Soc.* **54**, 481–518.

Chapman, C. H. and J. A. Orcutt (1985). Least-squares fitting of marine seismic refraction data, *Geophys. J. R. Astr. Soc.* **82**, 339–374.

Dunkin, J. W. (1965). Computation of modal solutions in layered elastic media at high frequencies, *Bull. Seism. Soc. Am.* **55**, 335–358.

Haskell, N. A. (1953). The dispersion of surface waves on multilayered media, *Bull. Seism. Soc. Am.* **43**, 17–34.

Kennett, B. L. N. (1981). Elastic wave propagation in stratified media, in *Advances in Applied Mechanics*, Chia-Shun Yih, Editor, Academic Press, New York, 79–167.

Kennett, B. L. N. (1983). *Seismic Wave Propagation in Stratified Media*, Cambridge University Press, Cambridge, England, 339 pp.

Kind, R. (1976). Computation of reflection coefficients for layered media. *J. Geophys.* **42**, 191–200.

Phinney, R. A., K. Roy Chowdhury, and L. N. Frazer (1981). Transformation and analysis of record sections, *J. Geophys. Res.* **86**, 359–377.

Shaw, P. R. and J. A. Orcutt (1985). Waveform inversion of seismic refraction data and applications to young Pacific crust, *Geophys. J. R. Astr. Soc.* **82**, 375–414.

DEPARTMENT OF GEOLOGICAL AND GEOPHYSICAL SCIENCES
PRINCETON UNIVERSITY
PRINCETON, NEW JERSEY 08544 (R.A.P., R.I.O.)

HAWAII INSTITUTE OF GEOPHYSICS
2525 CORREA ROAD
HONOLULU, HAWAII 96822 (G.J.F.)
CONTRIBUTION NO. 1852

Manuscript received 23 October 1985

Appendix

The interface matrix $\boldsymbol{F}_n$ is the matrix product

$$\boldsymbol{F}_n = \boldsymbol{T}_n^{-1}\boldsymbol{T}_{n+1}$$

where $\boldsymbol{T}_n$ is the 6×6 delta matrix for the nth layer. The elements of the Dunkin matrix are formed from the 2×2 subdeterminants of the 4×4 matrix which generates the stress-displacement vector from the wave potential vector. The interface matrix $\boldsymbol{F}_n$ is explicitly frequency-independent, and because we are interested only in ratios, we can ignore any explicit normalization, and common factors among the elements may be removed.

The following is a list of the 16 independent elements pf the Dunkin matrices. To reduce the number of multiplications required to form the elements, they have all been divided by μ.

$$t_{11} = -(p^2 + qq')/\mu = t_{16}$$
$$t_{12} = -2pq/\mu$$
$$t_{13} = -(p^2 - qq')/\mu = -t_{14}$$
$$t_{15} = -2pq'/\mu$$
$$t_{21} = iq'/\beta^2 = -t_{23} = -t_{24} = -t_{26}$$
$$t_{31} = -ip(\Gamma + 2qq') = t_{36} = t_{41} = t_{46}$$
$$t_{32} = -4ip^2q$$
$$t_{33} = -ip(\Gamma - 2qq') = t_{43} = -t_{34} = -t_{44}$$
$$t_{35} = -2i\Gamma q'$$
$$t_{42} = -2i\Gamma q$$
$$t_{45} = -4ip^2q'$$

$$
\begin{aligned}
&t_{51} = -iq/\beta^2 = t_{53} = t_{54} = -t_{56}\\
&t_{61} = -\mu(\Gamma^2 + 4p^2qq') = t_{66}\\
&t_{62} = -4\mu\Gamma pq\\
&t_{63} = -\mu(\Gamma^2 - 4p^2qq') = -t_{64}\\
&t_{65} = -4\mu\Gamma pq'\\
&t_{22} = t_{25} = t_{55} = t_{52} = 0
\end{aligned}
$$

where

$$\Gamma = 2p^2 - 1/\beta^2.$$

After removal of a common factor of qq', the inverse $\boldsymbol{T}_n^{-1}$ is seen to be just a rearrangement of the elements of $\boldsymbol{T}_n$

$$
\boldsymbol{T}^{-1} = \begin{bmatrix}
t_{61} & t_{51} & t_{31} & t_{31} & t_{21} & t_{11}\\
-t_{65} & 0 & -t_{45} & -t_{35} & 0 & -t_{15}\\
-t_{63} & -t_{51} & -t_{33} & -t_{33} & t_{21} & -t_{13}\\
t_{63} & -t_{51} & t_{33} & t_{33} & t_{21} & t_{13}\\
-t_{62} & 0 & -t_{42} & -t_{32} & 0 & -t_{12}\\
t_{61} & -t_{51} & t_{31} & t_{31} & -t_{21} & t_{11}
\end{bmatrix}.
$$